COFC

Cost Optimized Flow Control In Software Defined Networks

Banerjee, Anuradha; Hussain, Dil muhammed Akbar

Published in:
Procedia Computer Science

DOI (link to publication from Publisher):
10.1016/j.procs.2019.05.031

Publication date:
2019

Document Version
Publisher's PDF, also known as Version of record

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 152 (2019) 92–101

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Pervasive Computing Advances and Applications – PerCAA 2019

COFC: Cost Optimized Flow Control in Software Defined Networks

Anuradha Banerjee[a], D.M. Akbar Hussain[b] *

[a]*Dept of Computer Applications, Kalyani Govt. Engg. College, West Bengal, India*
[b]Dept. of Energy Technology, Aalborg University, Esbjerg, Denmark

Abstract

In a software defined network, there are various hosts and switches. Often multiple connections exist from one host to another through the switches. These connections require different amount of bandwidth as well as time delay. The proposed work Cost Optimized Flow Control (COFC) aims at finding the most efficient path between a pair of hosts and accordingly deactivates certain switches to let them go to sleep. This requires knowledge about topology of the network which is stored in a software-defined-network (SDN) controller. SDN decouples all the flow control activity from physical forwarding of packets. Flow control is entirely performed by the SDN controller as it determines the switches that can remain in sleep mode unless cost in its alternative paths crosses a threshold. This preserves energy and also optimizes the objectives bandwidth and delay.

Peer-review under responsibility of the scientific committee of the International Conference on Pervasive Computing Advances and Applications – PerCAA 2019.

*Keywords: Bandwidth; Cost; Delay; Energy Efficiency; Flow Control; Sleep; Software-defined-network Controller.*Abstract

1. Introduction

Software defined network (SDN) is a relatively new network paradigm that provides a suitable construct for simplifying network management and introducing real time programmability [1-4]. Ample scope of innovation is there. Flow control activity is handled by a centralized controller and it is completely decoupled from underlying hardware i.e. physical layer forwarding of packets. The Open Networking Foundation (ONF) [2], industry consortia defines the term SDN as “the physical separation of the network control plane from the forwarding plane and where

* Corresponding author. Tel.: +0-000-000-0000 ; fax: +0-000-000-0000 .
E-mail address: anuradha79bn@gmail.com

Peer-review under responsibility of the scientific committee of the International Conference on Pervasive Computing Advances and Applications – PerCAA 2019.
10.1016/j.procs.2019.05.031

a control plane controls several devices". High level of network abstraction is applied through the control plane, which consists of a centralized controller only. This eliminates the need of fine tuning hardware components to meet requirements of the network from time to time. SDN provides several application programming interfaces or API (both northbound and southbound) to execute routing, security and access control.

The present article proposes one cost based flow control mechanism where optimized path between any two hosts in the network is determined based on bandwidth, estimated initial packet drop rate without wait in flow queue or message queue of a switch, timestamp of dropping the first packet and delay in composite links. If there are some switches that do not exist in the entire collection of optimum paths, then they are deactivated. They can remain in sleep mode unless software interrupts from the SDN controller raise them up again. Exactly one software interrupt is required to activate each switch. The interrupt is raised if any one switch suffers from shortage of battery power. In simulation section, COFC is compared with its state-of-the-art competitors and results show significant improvement in favor of COFC.

As far as the novelty of COFC is concerned, the idea is simple but effective and innovative. This technique is applicable for SDNs of various kinds (simple as well as complex topologies) because COFC requires knowledge of all available paths between all pairs of source and destination hosts, bandwidth of links, average delay in switches along with their rate of dropping packets etc. This is possible only for SDN because SDN decouples physical packet forwarding from control plane. Packets flow as per instruction of the controller. To the best of authors knowledge, no similar work exists in the literature of SDN.

2. Background and Related Work

An important characteristic of dense software defined networks is rich link connectivity that is, one host is connected to other host through multiple different paths. Statistically, it has been shown that network traffic peaks during day time and falls at night. This leaves most of links in idle but on state consuming a huge amount of energy.

An energy-aware routing method is proposed in [15] that tries to use as few network devices as possible to take the responsibility of routing from one host to another. The strategy proposed in [11] is based on ternary content-addressable memory (TCAM) that is placed inside switches [11]. If TCAM capacity of a switch is 0, then the switch is deactivated and no traffic can pass through it. However, it does not take care of bandwidth, packet drop rate, delay etc. We shall refer to this work as TCAM-SLP in rest of the paper. ElasticTree [17] is presented by Heller et. Al., that dynamically adjusts the set of active network elements to change traffic in a data center of Fat-Tree topology. CARPO [13] is a power optimization algorithm that applies correlation analysis among multiple flows and traffic is consolidated for energy saving provided link capacity permits. A network function virtualization based technique is applied for energy efficiency in [20].

ElastiCon [18] is a distributed controller architecture in which a pool of controllers is there. As per various traffic conditions, a controller is elected to serve in the control plane. Load is dynamically shifted from one controller to another. Fu et. Al. [14], proposed a dormant multi-controller model that applies quantitative analysis of energy consumption as well as performance throughput of multiple controllers. E^3MC [19] is another multi-controller system that applies energy efficiency using SDN. However, none of these energy saving mechanisms consider bandwidth and delay. Also they do not estimate packet drop rate in the links. The present work COFC takes care of these before selecting one particular host-to-host communication path.

3. System Modeling

The underlying network consists of certain hosts and switches. It is expressed as a graph G = (CNTL, H, S, LN) where CNTL is the special controller node or vertex. H denotes the set of hosts and S stands for the set of switches. LN is the set of links from vertex a to vertex b in graph G such that the following conditions hold:

i) $a \in H$ and $b \in S$

ii) $a \in S$ and $b \in H$

iii) $a \in S$ and $b \in S$

Assume that $path_{ij}$ denotes the set of different paths from one host h_i to another host h_j, s.t. $h_i, h_j \in H$. If k number of paths exist from h_i to h_j, then,

$$path_{ij} = \{p_{ij}(1), p_{ij}(2), \dots, p_{ij}(k)\}$$

Where $p_{ij}(v)$ s.t. $1 \le v \le k$, denote v-th path from h_i to h_j.

Application of COFC will be possible for a network provided the following condition CND is satisfied.

$$\underline{CND::} \sum_{\substack{h_i, h_j \in H, \\ i \neq j}} |path_{ij}| > 1$$

The condition CND specifies that more than one path exists for at least one pair of distinct hosts. In that case, comparison between various paths between same pair of source and destination hosts, is possible and finding one optimum out of the available options, become relevant. For example, if we assume that between two hosts h_i and h_j, following three paths exist: Path1(i,j) = {h_i->sw1->sw3->sw4->sw7->h_j},Path2(i,j) = {h_i->sw1->sw5->sw8->sw7->h_j},Path3(i,j) = {h_i->sw1->sw6->sw2->sw7->h_j}. If we assume that Path1(i,j) is chosen as optimal, then it will open up the possibility of turning off the switches sw2, sw5, sw6 and sw8. But if the three paths are as below, then only the switches sw2 and sw8 may be turned off, because Path2(i,j) and Path3(i,j) have so many nodes in common. Path1(i,j) = {h_i->sw1->sw3->sw4->sw7->h_j} Path2(i,j) = {h_i->sw1->sw4->sw8->sw7->h_j} Path3(i,j) = {h_i->sw1->sw3->sw2->sw7->h_j}. Therefore, initial applicability IA(NT) of a network topology NT is formulated in (1). It is based on the concept that, if a large number of disjoint paths exist between various source-destination host pairs, then a good number of non-ingress switches may be turned off without hampering vertex to vertex communication. That is, size of the set LN is expected to get reduced up to a great extent after applying COFC on network topology NT.

$$IA(NT) = \max_{\substack{h_i, h_j \in H, \\ i \neq j}} \{ disjoint_{ij} \} / (|S| - |H|) \quad (1)$$

$$disjoint_{ij} = [\sum_{\substack{1 \le u,v \le |path_{ij}|, \\ u \neq v}} \{ |p_{ij}(u) - p_{ij}(v)| \}] / |path_{ij}| \quad (2)$$

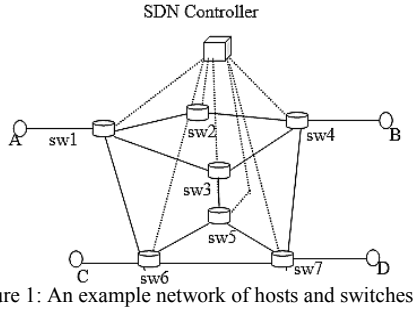

Figure 1: An example network of hosts and switches

From the formulation in (2) we understand that $disjoint_{ij}$ is the average number of switches that can be turned off provided one path is selected to be the optimum out of $|path_{ij}|$ number of paths. Each path option is checked for disjointness with rest of the options between the same pair of source-destination hosts. Please note that maximum possible number of switches that can be turned off, is given by (|S| - |H|), because (|S| - |H|) is the total number of non-ingress switches. Possibility of turning all of them off, arises only when in the optimal path, ingress switches of source and destination hosts, directly communicate with each other. From the formulation (1) it is clear that, IA(NT) lies between 0 and 1. Application of COFC is performed on a network topology provided IA(NT) is greater than 0.25. Assume that for the network topology NT, Z(NT) denote the set of switches can remain deactivated for some time. For each switch $swch \in Z(NT)$, let deactivation time duration be $\tau(swch)$ and rate of energy consumption in the same switch, be $\varpi(swch)$. If overall simulation time is Y, then energy E(NT) saved through deactivation of those switches belonging to the set Z(NT), is formulated in (3).

$$E(NT) = \sum_{swch \in Z(NT)} (\tau(swch) \times \varpi(swch)) \quad (3)$$

One such example is shown in figure 1. We can see that two different paths exist from A to B. They are as follows:

$path1_{A \to B}$: A→sw1→sw2→sw4→B

$path2_{A \to B}$: A→sw1→sw3→sw4→B

Each link in path1 and path2 require different amount of bandwidth. Also the delay they suffer is not expected to be uniform. COFC estimates the cost of each path in terms of bandwidth, approximate initial packet drop rate without wait in flow queue or message queue of a switch and delay, while background remains the original network in Figure 1. The option incurring minimum cost is elected for communication from host A to B. Without any loss of generality, we assume that cost incurred in $path2_{A \to B}$ is less than the same in $path1_{A \to B}$ in context of overall network in Figure 1. Therefore $path2_{A \to B}$ will be chosen for communication from A to B.

The network is modeled as a graph G = (V, E) where V is the set of vertices and E is the set of edges. By vertices we mean hosts and switches (subsequently called as nodes in general) whereas edge represents a connection between any two vertices. COFC finds a cost optimized version of G based on bandwidth, approximate initial packet drop rate without wait in flow queue or message queue of a switch and delay in the links. Let G′ = (V′, E′) denote the reduced G, where V′=V-SL s.t. SL is the set of sleeping switches and E′ represents the set of connections in cost optimized network graph G′. Corresponding to the network in Figure 1, G and G′ are shown in Figures 2 and 3, respectively, based on optimum bidirectional path assumptions in Table 1.

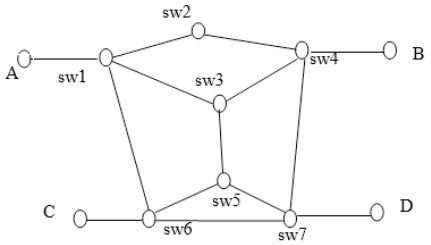

Figure 2: G corresponding to the network in Figure 1

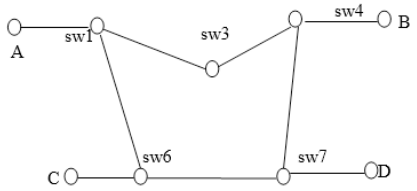

Figure 3: G′ corresponding to the graph of Figure 2

Table 1: Optimum paths from host-to-host

Source Host	Destination Host	Path
A	B	A→sw1→sw3→sw4→B
A	C	A→sw1→sw6→C
A	D	A→sw1→sw6→sw7→B
B	C	B→sw4→sw7→sw6→C
B	D	B→sw4→sw7→D
C	D	C→sw6→sw7→D

SL = {sw2, sw5}

The switches that can be deactivated in fig. 2 are sw2 and sw5.

4. SDN Controller Design

4.1 Overview

The SDN controller is equipped with four different tables, named host-call table, switch-dep-drop table, link-data table and min-cost-path table. Attributes of host-call table are as follows:

i) host-id

ii) max-call-gen

host-id is the unique identification number of a host. For example, in the graph G of figure 2, host-ids are A, B, C and D. The other attribute max-call-gen is the maximum call generation rate of the host, that is, maximum number of calls generated by that host per unit time.

Attributes of switch-dep-drop table are mentioned below:

i) switch-id
ii) call-dep-rate
iii) queue-size
iv) init-pac-drop-rate-no-wait
v) first-drop-time

switch-id specifies unique identification number of the switch. As far as creation of tables are concerned, whenever any host registers into the network, its maximum call generation rate is available to the controller and the controller itself assigns an unique host identifier to that new host. Then the entry (host-id, maximum call generation rate) are inserted into the host-call table. As far as registering new switches into the network is concerned, information like switch-id, maximum possible call servicing rate or call-dep-rate and queue-size are known to the SDN controller. From these information initial packet drop rate without wait in flow queue or message queue of a switch (init-pac-drop-rate-no-wait) and minimum time duration between starting operation in the network and dropping the first packet (that is, first-drop-time) can be calculated. For example, suppose, one particular switch swch is associated with a set of hosts denoted by H1(swch) and a set of switches S1(swch). Also assume that, maximum call generation rate of a host h_i is m(i) and size of flow queue or message queue of swch is f(swch). Call departure rate of a switch swch is c(swch). Therefore, maximum initial packet drop rate without wait in flow queue or message queue of a switch in swch is denoted by mpdr(swch) and defined in (4).

$$mpdr(swch) = \sum_{h_i \in H1(swch)} m(i) + \sum_{sw \in S1(swch)} c(sw) - c(swch) \quad (4)$$

From (4), first drop time fdm(swch) of a switch swch is computed in (5).

$$fdm(swch) = \lfloor f(swch)/ mpdr(swch) \rfloor \quad (5)$$

For example, in the graph G of figure 2, switch-ids are sw1, sw2, sw3, sw4, sw5, sw6 and sw7. call-dep-rate is the fixed call departure rate of a switch. Each switch is equipped with a message queue having different sizes. first-drop-time is the timestamp of dropping the first packet. init-pac-drop-rate-no-wait is the approximate initial packet drop rate without wait in flow queue or message queue of the switch. It is computed as follows, considering network topology graph G in fig. 2. In that graph, sw1 is connected to the network elements sw2, sw3, sw6 and A. Among these, only A can generate calls while the switches sw2, sw3 and sw6 can only forward. If max-call-gen of A is 0.3 (A can generate 0.3 calls per unit time) and call-dep-rates of sw2, sw3 and sw6 are 0.05, 0.7 and 0.3, then maximum number of calls that can arrive at the message queue of sw1, is given by (0.3+0.05+0.7+0.3) i.e. 1.35. If call-dep-rate of sw1 is 2, then init-pac-drop-rate-no-wait of sw1 will be 0 irrespective of the size of its message queue. On the other hand, if call-dep-rate of sw1 is 0.5, then rate of dropping packets in sw1 will be (1.35 – 0.5) i.e. 0.85, irrespective of queue size. Assuming that size of message queue of sw1 be 3, timestamp of first packet drop or first-drop-time is given by, $\lfloor$(queue-size / init-pac-drop-rate-no-wait)$\rfloor$ i.e. $\lfloor(3/0.85)\rfloor$ or $\lfloor 3.529 \rfloor$ or 3.

link-data table contains the bandwidth and delay information in each table. Names of the attributes are link-id (specifying identification numbers of endpoints of a link), link-bandwidth and link-delay. min-cost-path table consists of minimum cost path information from each host to other. It's attributes are start-host-id, end-host-id, path, cost and opt-path. start-host-id and end-host-id are host identifiers at beginning and end of a path. path specifies all possible router sequences in between those two hosts and cost is the overall associated cost involved in that path. As the name implies, opt-path is the optimum path in between those two hosts. Cost is computed based on information in the three other tables present in SDN controller. Assume that between any two hosts h_i and h_j, path $p_{ij}(u)$ is chosen as optimal where $1 \le u \le |path_{ij}|$, where $p_{ij}(u)$ is given by, $p_{ij}(u) = \{h_i \text{->} swch_q \text{->} swch_{q+1} \text{->} swch_{q+2} \text{->} \dots \text{->} swch_{q+g} \text{->} h_j\}$. Here $swch_q$ is the ingress switch of h_i whereas $swch_{q+g}$ is the same of h_j. Then, overall bandwidth $BW(p_{ij}(u))$ of the path $p_{ij}(u)$ is formulated in (4) whereas overall delay $DL(p_{ij}(u))$, initial packet drop rate without wait in flow queue or message queue of a switch $PDR(p_{ij}(u))$ and average first drop time $AFDM(p_{ij}(u))$ are given by (7), (8) and (9) respectively.

$$BW(p_{ij}(u)) = bw(h_i \text{->} swch_q) + \sum_{1 \le v \le (g-1)} bw(swch_{v+q} \text{->} swch_{v+q+1}) + bw(swch_{q+g} \text{->} h_j) \quad (6)$$

$$DL(p_{ij}(u)) = dl(h_i \text{->} swch_q) + \sum_{1 \le v \le (g-1)} dl(swch_{v+q} \text{->} swch_{v+q+1}) + dl(swch_{q+g} \text{->} h_j) \quad (7)$$

$$PDR(p_{ij}(u)) = \sum_{q \le v \le (g+q)} mpdr(swch_v) \quad (8)$$

$$AFDM(p_{ij}(u)) = [\sum fdm(swch_v)] / (g+1) \quad (9)$$

$q \le v \le (g+q)$

Here bw(a->b), dl(a->b), pdr(a) and fdm(a) indicate bandwidth and delay of the link from a to b, initial packet drop rate without wait in flow queue or message queue of a switch and first drop time of the switch a, respectively. Then overall cost $C(p_{ij}(u))$ of the path $p_{ij}(u)$ is given by (8).

$$C(p_{ij}(u)) = (BW(p_{ij}(u)) \times DL(p_{ij}(u)) \times DL(p_{ij}(u)) \times AFDM(p_{ij}(u))) \quad (10)$$

Cost saved in COFC by selecting $p_{ij}(u)$ as optimal path as per the logic of COFC, is denoted as $s\text{-}c(p_{ij}(u))$ and formulated in (11). The set of other path options between the same pair of hosts is given by, $(path_{ij} - p_{ij}(u))$. Cost of each path $pth \in (path_{ij} - p_{ij}(u))$, cost saved is given by, $(C(pth) - C(p_{ij}(u)))$. Overall average cost saved by choosing $p_{ij}(u)$ as optimal, is denoted by $acs(p_{ij}(u))$ and defined in (11).

$$acs(p_{ij}(u)) = [\{\sum_{pth \in (path_{ij} - p_{ij}(u))} (1 - C(p_{ij}(u))/C(pth))\} /(|path_{ij}| - 1)] \times 100 \quad (11)$$

Average cost saved in percentage throughout the simulation of COFC, is denoted as PSPF. Let F specify the total set of flows and for each flow f, let opt(f) denote the optimal path. Then, PSPF is formulated in (12).

$$PSPF = \sum_{f \in F} acs(opt(f)) / |F| \quad (12)$$

4.2 Illustration with an Example

Assume that corresponding to the graph G of Figure 2, the following information is maintained in Table 2, Table 3 and Table 4.

Table 2: host-call table of G

host-id	max-call-gen
A	0.3
B	0.2
C	0.5
D	0.1

Table 3: switch-dep-drop table of G

switch-id	call-dep-rate	queue-size	init-pac-drop-rate-no-wait	first-drop-time
sw1	2	10	0	0
sw2	0.05	8	2.04	3
sw3	0.7	12	2.4	5
sw4	0.9	8	0.05	160
sw5	0.2	12	1.2	10
sw6	0.3	8	0.8	10
sw7	0.4	10	0.2	50

Table 4: Link data information

link-id	link-bandwidth	link-delay
(A,sw1)	2	1
(sw1, sw6)	1	3
(C, sw6)	5	2
(sw1, sw3)	1	2
(sw3, sw5)	3	2
(sw5, sw6)	4	1
(sw5, sw7)	3	1
(sw6, sw7)	1	1
(sw1, sw2)	10	10
(sw2, sw4)	10	2
(sw4, sw3)	2	2
(sw4, B)	1	1
(sw4, sw7)	4	2
(sw7, D)	1	1

Table 5: min-cost-path table

start-host-id	end-host-id	Path	cost	opt-path

A	B	A → sw1 → sw2 → sw4 → B	16.52	A → sw1→ sw6→sw7 →sw4 → B
		A → sw1 → sw3 → sw4 → B	2.140	
		A → sw1 → sw6 → sw5 → sw3→sw4 → B	35.34	
		A → sw1→ sw3 → sw5 → sw7 →sw4 → B	12.94	
		A → sw1→ sw3 → sw5 → sw6→sw7 →sw4 → B	19.95	
		A → sw1→ sw6→sw7 →sw4 → B	**1.718**	
		A → sw1→ sw6→sw5→sw7 →sw4 → B	7.923	
A	C	**A → sw1 → sw6 → C**	**11.52**	A → sw1→ sw6→C
		A → sw1 → sw3 → sw5 → sw6→C	105.6	
		A → sw1 → sw3 → sw5 → sw7→sw6 → C	38.88	
		A → sw1→ sw3 → sw4 → sw7 →sw6 → C	13.8	
		A → sw1→ sw3 → sw4→ sw7→sw5 →sw6 → C	31.99	
		A → sw1→ sw2→sw4 →sw7 → sw6→C	47.89	
		A → sw1→ sw2→sw4→sw7 →sw5 → sw6→C	93.05	
		A → sw1→ sw2→sw4→sw3→sw5 → sw6→C	17.39	
		A → sw1→ sw2→sw4→sw3→sw5 → sw7→sw6→C	170	
A	D	**A → sw1 → sw6 → sw7→D**	**2**	A → sw1→ sw6→sw7→D
		A → sw1 → sw3 → sw5 → sw7→D	20.46	
		A → sw1 → sw3 → sw5 → sw6→sw7 → D	35.33	
		A → sw1 → sw2 → sw4 → sw7→D	23.22	
		A → sw1→ sw2 → sw4→ sw3→sw5 →sw7 → D	106.5	
		A → sw1→ sw2 → sw4→ sw3→sw5 →sw6→sw7 → D	148.42	
		A → sw1→ sw6→sw5→sw7 →D	30.8	
		A → sw1→ sw6→sw5→sw3→sw4 → sw7→D	28.26	
B	C	B → sw4 → sw2 → sw1 → sw6→C	40.59	B → sw4→ sw7 → sw6→ C
		B → sw4 → sw2 → sw1 → sw3 → sw5 → sw6 → C	164.32	
		B → sw4 → sw2 → sw1 → sw3 → sw5 → sw7→ sw6 → C	160.56	
		B → sw4→ sw7 → sw6→ C	**0.945**	
		B → sw4→ sw7 → sw5 →sw6→ C	14.43	
		B → sw4→ sw3 → sw5→sw6→ C	5.82	
		B → sw4→ sw3 → sw5→sw7→sw6→ C	16.02	
		B → sw4→ sw3 → sw1→sw6→ C	9.28	
B	D	**B → sw4 → sw7→D**	**0.0857**	B → sw4→ sw7 → D
		B→ sw4 → sw3 → sw5 → sw7→D	5.99	
		B→ sw4→ sw3 → sw5 → sw6→sw7 → D	11.397	
		B→ sw4 → sw2 → sw1 → sw3 → sw5 → sw7→ D	99.13	
		B→ sw4 → sw2 → sw1 → sw3 → sw5 → sw6→sw7→ D	139.42	
		B→ sw4 → sw2 → sw1 → sw6 → sw7→ D	40.10	
		B→ sw4 → sw2 → sw1 → sw6→ sw5→sw7→ D	72.84	
		B→ sw4 → sw3 → sw1 → sw6→ sw5→sw7→ D	19.81	
		B→ sw4 → sw3 → sw1 → sw6→sw7→ D	6.44	
C	D	**C → sw6 → sw7→D**	1.4	C→ sw6 → sw7→D
		C→ sw6 → sw5 → sw7→D	8.17	
		C→ sw6→ sw5 → sw3 → sw4→sw7 → D	22.85	
		C→ sw6→ sw5 → sw3 → sw1 →sw2 →sw4→sw7 → D	187.99	
		C→ sw6→ sw1 →sw3 →sw4→sw7 → D	15.59	
		C→ sw6→ sw1 →sw3 →sw5→sw7 → D	51.52	
		C→ sw6→ sw1 →sw2 →sw4→sw7 → D	54.67	
		C→ sw6→ sw1 →sw2 →sw4→ sw3 →sw5→sw7 → D	173.15	

Based on the above information, minimum cost paths from one host to another, are computed and stored in min-cost-path table.

Suppose there is a path PT from A to B in graph G of fig. 2, such that, PT = A → sw1 → sw3 → sw4 → B. As per table 2, overall bandwidth of the path is {link-bandwidth (A → sw1) + link-bandwidth(sw1 → sw3) + link-bandwidth(sw3 → sw4) + link-bandwidth(sw3 → sw4) + link-bandwidth(sw4 → B) }i.e. (2+1+2+1) or 6. Similarly, overall delay of the path is, {delay (A → sw1) + delay(sw1 → sw3) + delay(sw3 → sw4) + delay(sw4 →

B) }i.e. (1+2+2+1) or 6. Overall init-pac-drop-rate-no-wait of PT is due to only the switches, i.e. sw1, sw3 and sw4; hosts cannot drop packets. Therefore overall init-pac-drop-rate-no-wait of PT is {init-pac-drop-rate-no-wait(sw1)+init-pac-drop-rate-no-wait(sw3)+init-pac-drop-rate-no-wait(sw4)} i.e. (0+2.4+.05) or 2.45. Average first-drop-time is summation of first-drop-time of all the switches divided by number of switches. For the path PT, average first-drop-time is (0+5+160)/3 i.e. 165/3. Cost C(PT) of the path PT, is thus formulated in (13).

$$C(PT) = \{(6\times6\times2.45)/(165/3)\} = 2.853 \quad (13)$$

In this way, costs of all paths between all pairs of hosts are computed and the path with minimum cost between each pair of hosts, is selected for communication. The switches that do not appear in any of these optimal paths, can be switched off to save energy consumption in the network. Corresponding G′ appears in Figure 4.

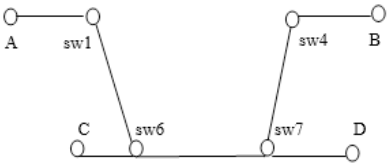

Figure 4: Reduced network graph after deactivating possible switches

SL = {sw2, sw3, sw5}

5. Simulation Environment And Results

5.1 Simulation Environment

Simulation is run using MATLAB 2015b on a server with 3.1GHz Intel Core 2 Duo Processor, 16 GB RAM and Windows 8.1. State-of-the-art competitors of COFC are TCAM-SLP and CARPO. Numbers of switches are 10, 30, 50, 70 and 100. Number of hosts is kept constant at 25. Number of flows vary as 10, 100, 1000, 10000, 100000. Number of simulation runs is 10. The performance metrics are as follows

i) Percentage of energy saved per flow (PEPF)

ii) Percentage of saving in cost per flow (PSPF)

These are measured with respect to number of switches and number of flows. Formulation of PEPF appear in (16). PSPF is already defined in (12). When numbers of switches are varied, number of flows is kept constant at 1000. Similarly when measured with respect to number of flows, number of switches are kept constant at 70.

$$PEPF = \{(A1 - B1) / (A1\times flowcnt)\} \times 100$$

$$A1 = [\sum_{h_i\in H} e\text{-}c(h_i)_{Without\text{-}COFC} + \sum_{swch\in S} e\text{-}c(swch)_{Without\text{-}COFC})] \quad (14)$$

$$B1 = [\sum_{h_i\in H} e\text{-}c(h_i)_{COFC} + \sum_{swch\in S} e\text{-}c(swch)_{COFC})] \quad (15)$$

$$PEPF = [(\sum_{h_i\in H} e\text{-}c(h_i)_{Without\text{-}COFC} + \sum_{swch\in S} e\text{-}c(swch)_{Without\text{-}COFC}) - (\sum_{h_i\in H} e\text{-}c(h_i)_{COFC} + \sum_{swch\in S} e\text{-}c(swch)_{COFC})]/flowcnt \quad (16)$$

$e\text{-}c(h_i)_{COFC}$ and $e\text{-}c(swch)_{COFC}$ denote energy consumed by host h_i and switch swch in COFC throughout the simulation period whereas $e\text{-}c(h_i)_{Without\text{-}COFC}$ and $e\text{-}c(swch)_{Without\text{-}COFC}$ denote energy consumed by host h_i and switch swch in a non-COFC communication throughout the simulation period. flowcnt is the total number of flows throughout the simulation.

5.2 Simulation Results

With increase in number of network switches, a lot of new routes evolve between any two pair of hosts. Therefore, the difference between energy consumption corresponding to any two routes between any two pair of hosts generally increases or remains the same in some cases but never decreases. TCAM-SLP deactivates a switch if its TCAM capacity is 0. CARPO utilizes correlation between packets. However none of them focus on the rate of dropping packets by switches belonging to one particular path. Packets that are dropped have to be resent sometime later again, consuming more energy in those paths. Similarly the time delay before dropping the first packet in a

path also matters. If average first-drop-time of a path is high, then it may happen that the entire session of communication completes before the first packet through the route can be dropped. All these contribute to energy efficiency of COFC than its state-of-the-art competitors. This is shown in Figure 5.

Figure 6 clearly illustrates the fact that when network load is low, COFC significantly saves energy than its competitors. For example, when network flow is close to 100, almost 86% energy is saved. The figure goes down to 42% when number of flows is as high as 100000. As expected, energy efficiency effectiveness decreases as network load increases. As already mentioned, cost of a path is computed as, cost = {bandwidth×delay×init-pac-drop-rate-no-wait×avg(first-drop-time)}. These factors, specially bandwidth, delay and init-pac-drop-rate-no-wait are very important constituents of cost of a path. These are neither considered by TCAM-SLP, nor by CARPO. With increase in number of network switches, new route options open between pairs of hosts. The links have different bandwidth, delay etc. This saves more cost, as shown in Figure 7.

As number of flows in the network increase, it may happen that after some time a switch in the optimum path runs out of the battery power and the link breaks. In that case, new optimum will be the next choice in min-cost-path table, corresponding to the same pair of source and destination nodes. As a result, cost saving decreases. Still, since COFC elects the optimum path based on important factors like bandwidth, delay, pac-dop-rate etc., so COFC produces substantial improvement in terms of cost, than its competitors. This is evident from Figure 8.

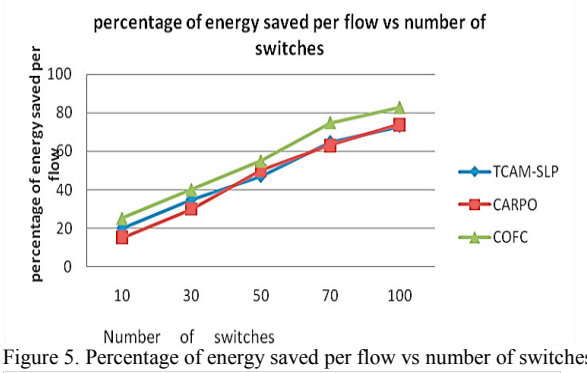

Figure 5. Percentage of energy saved per flow vs number of switches

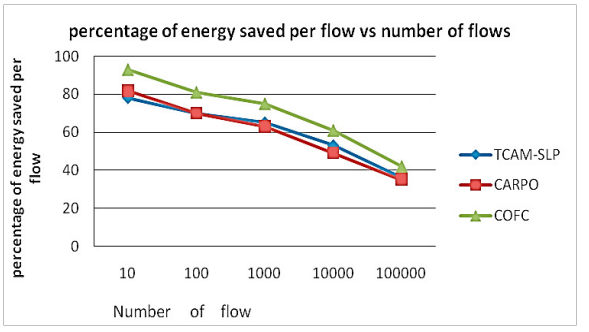

Figure 6: Percentage of energy saved per flow vs number of flows

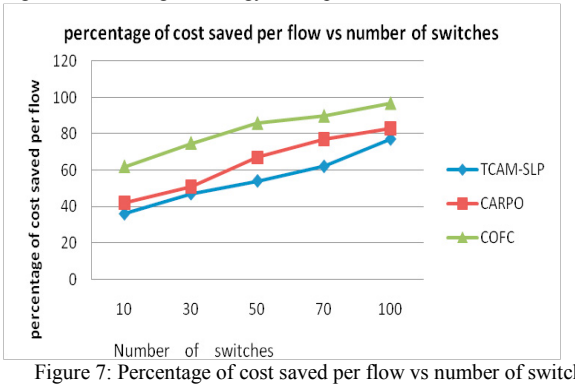

Figure 7: Percentage of cost saved per flow vs number of switches

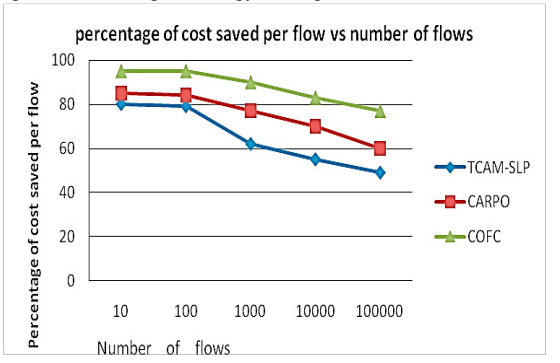

Figure 8: percentage of cost saved per flow vs number of flows

6. Conclusion

The main strength of COFC is the estimation of cost of multiple routes between various pairs of hosts. In a software defined network, routes are always established through SDN controller. Main advantage of this centralized configuration is that SDN controller can be made aware of topology of the whole network. If different aspects of cost like bandwidth, delay, estimated initial packet drop rate without wait in flow queue or message queue of a switch, approximate time before dropping of first packet , etc, are not considered then the problem would come down to simple shortest path problem in a graph. The present article COFC demonstrates importance of various cost components in determining overall performance of a route and its claims are strongly supported by encouraging

simulation results. The results show that COFC produces great energy saving by deactivating certain switches. It includes those links in a path that do not consume much bandwidth, and do not produce much delay.

7. Future Scope

Our aim is to modify COFC in such a manner that will not overload SDN controller in case of a huge number of hosts and switches. If the network becomes very big, then a pool of controllers can be employed. Each controller will be aware of topology of certain zones in the network. The entire network can be divided into some hexagonal regions called zones. Each zone will consist of some hosts and switches. Certain peripheral switches will act as inter-zone routers. This distributed architecture will divide route-selection load among multiple controllers and save a lot of energy in the system.

References

[1] T. Bakhshi. (2017) "State of the art and Recent Research Advances in Software Defined Networking," *Wireless Communications and Mobile Computing.*

[2] SDN Architecture, Issue 1, Open Networking Foundation, https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR SDN ARCH 1.0 060620- 14.pdf., 2014

[3] I. F. Akyildiz, A. Lee, P.Wang, M. Luo, and W. Chou. (2014) "A roadmap for traffic engineering in SDN-OpenFlow networks," *Computer Networks*, **vol. 71**, pp. 1–30.

[4] N. Feamster, J. Rexford, and E. Zegura. (2014) "The road to SDN: an intellectual history of programmable networks," *ACM SIGCOMM Computer Communication Review*, **vol. 44, no. 2**, pp. 87– 98.

[5] N. McKeown, T. Anderson, H. Balakrishnan et al. (2008) "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, **vol. 38, no. 2**, pp. 69–74.

[6] Open Networking Foundation (ONF), https://www.opennetworking.org/.

[7] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie. (2015) "A survey on software-defined networking," *IEEE Communications Surveys & Tutorials*, **vol. 17, no. 1**, pp. 27–51.

[8] B. A. A. Nunes, M. Mendonca, X.N. Nguyen, K. Obraczka, and T. Turletti. (2014) "A survey of software-defined networking: past, present, and future of programmable networks," *IEEE Communications Surveys & Tutorials*, **vol. 16, no. 3**, pp. 1617– 1634.

[9] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig. (2015) "Software-defined networking: a comprehensive survey," *Proceedings of the IEEE*, **vol. 103, no. 1**, pp. 14–76.

[10] A. Banerjee, F. Esposito. (2017) "A Survey of Scheduling Policies in Software Defined Networks," *In Proceedings of IEEE ANTS.*

[11] S. Jadala, S. Pelluri. (2017) "Energy Optimization At Cloud Data Centre Using SDN," *International Journal of Engineering Trends and Technology.*

[12] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker and J. Turner. (2008) "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communications Review*, **vol. 38.**

[13] X. Wang, Y. Yao, K. Lu, Q. Cao. (2012) "Carpo: Correlation-aware power consumption in data centre network", *In Proceedings of IEEE INFOCOM.*

[14] A. Bavier, N. Feamster, M. Huang, L. Peterson, and J. Rexford. (2006) "In VINI veritas: realistic and controlled network experimentation," *ACM SIGCOMM Computer Communication Review*, **vol. 36, no. 4**, pp. 3–14.

[15] http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=63A5623CBB2096EF4DFD4919B900584B?doi=10.1.1.476.3036&rep=rep1&type=pdf

[16] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu. (2009) "BCube: A High Performance, Server-centric Network Architecture for Modular Data Centers," *In ACM SIGCOMM*.

[17] B. Heller et. al. (2010) "Elastic Tree: Saving energy in data center networks," *in Proceedings of USENIX NSDI*.

[18] A.Dixit et. al. (2014) "ElastiCon: An elastic distributed SDN controller," *In Proceedings of 10th IEEE/ACM Symp. Archit. Netw. Commun. Syst*.

[19] K. Xie et. al. (2016) "E^3MC: Improving Energy Efficiency via Elastic Multi-Controller SDN in Data Center Networks," *IEEE Access.*

[20] Q. Duan, N. Ansari, M. Toy. (2016) "Software Defined Network Virtualization: an Architectural Framework for Integrating SDN and NFV for Service Provisioning in Future Networks," *IEEE Network magazine*, **vol. 30, no. 5**, pp. 10-16.